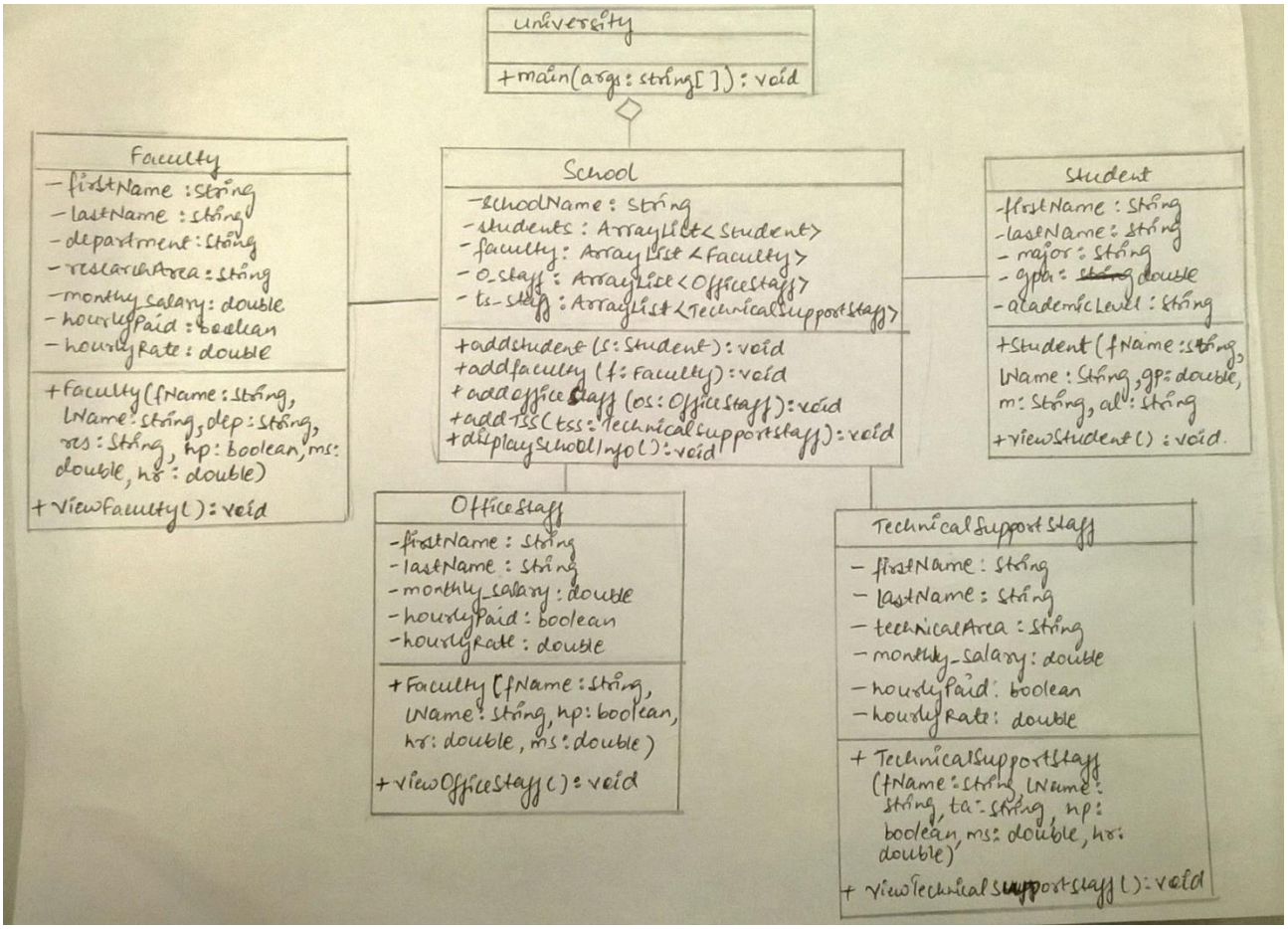
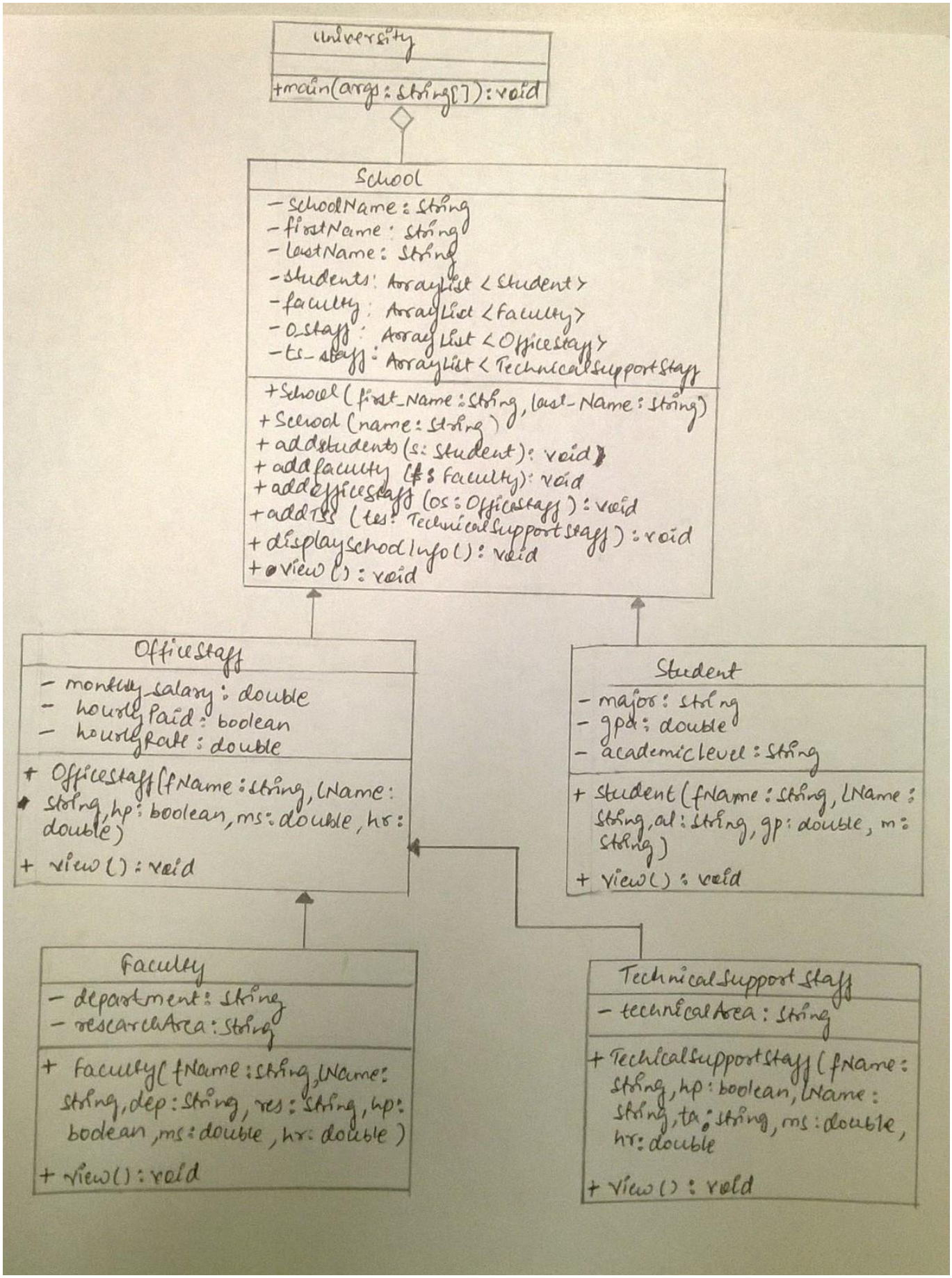
1) a)



b) The current program has a lot of redundancy in the code. The data attributes firstName and lastName are used in four classes for storing the name of students and staff. Data attributes like hourlyPaid, hourlyRate and monthly_salary are used in three classes for doing a common function in all. Also, view function in every class is performing a common task of displaying some part which is creating redundancy in code. Thus, the current code is violating a lot of OO design issues and in order to get effective internal structure, the code should be refactored so that output remains the same and reducing repetition of the code.

The code can be refactored by implementing object oriented design principles such as **polymorphism, inheritance and data abstraction**. Data abstraction can be implemented by defining the firstName and lastName as common variable in a super class (School class) as protected and later use them in the base class using the appropriate keyword. In the same way, monthly_salary, hourlyPaid, and hourlyRate can be defined in an OfficeStaff class as a protected data member and later can be used in the Faculty and TechnicalSupportStaff classes using the appropriate keyword. Polymorphism can be implemented by creating a common view method in the super class (School.java) and calling this method with same name from every base class instead of creating it separately in every class. All these techniques will change the internal structure of the code and making it more efficient but the output will remain same. Thus, in the refactored code, School class will be the super class with OfficeStaff class and Student class as the base class and OfficeStaff class as the super class for Faculty and TechinicalSupportStaff classes.

c)


University
+main(args: String[]): void
School
- schoolName: String
- firstName: String
- lastName: String
- students: ArrayList<Student>
- faculty: ArrayList<Faculty>
- o_staff: ArrayList<OfficeStaff>
- ts_staff: ArrayList<TechnicalSupportStaff>
+School(first_Name: String, last_Name: String)
+School(name: String)
+addStudents(s: Student): void
+addFaculty(f: Faculty): void
+addOfficeStaff(os: OfficeStaff): void
+addTSS(tss: TechnicalSupportStaff): void
+displaySchoolInfo(): void
+view(): void
OfficeStaff
- monthly_salary: double
- hourlyPaid: boolean
- hourlyRate: double
+ OfficeStaff(fName: String, LName: String, hp: boolean, ms: double, hr: double)
+ view(): void
Student
- major: String
- gpa: double
- academicLevel: String
+ Student(fName: String, LName: String, al: String, gp: double, m: String)
+ view(): void
Faculty
- department: String
- researchArea: String
+ Faculty(fName: String, LName: String, dep: String, res: String, hp: boolean, ms: double, hr: double)
+ view(): void
TechnicalSupportStaff
- technicalArea: String
+ TechicalSupportStaff(fName: String, hp: boolean, LName: String, ta: String, ms: double, hr: double
+ View(): void